Fig. 1

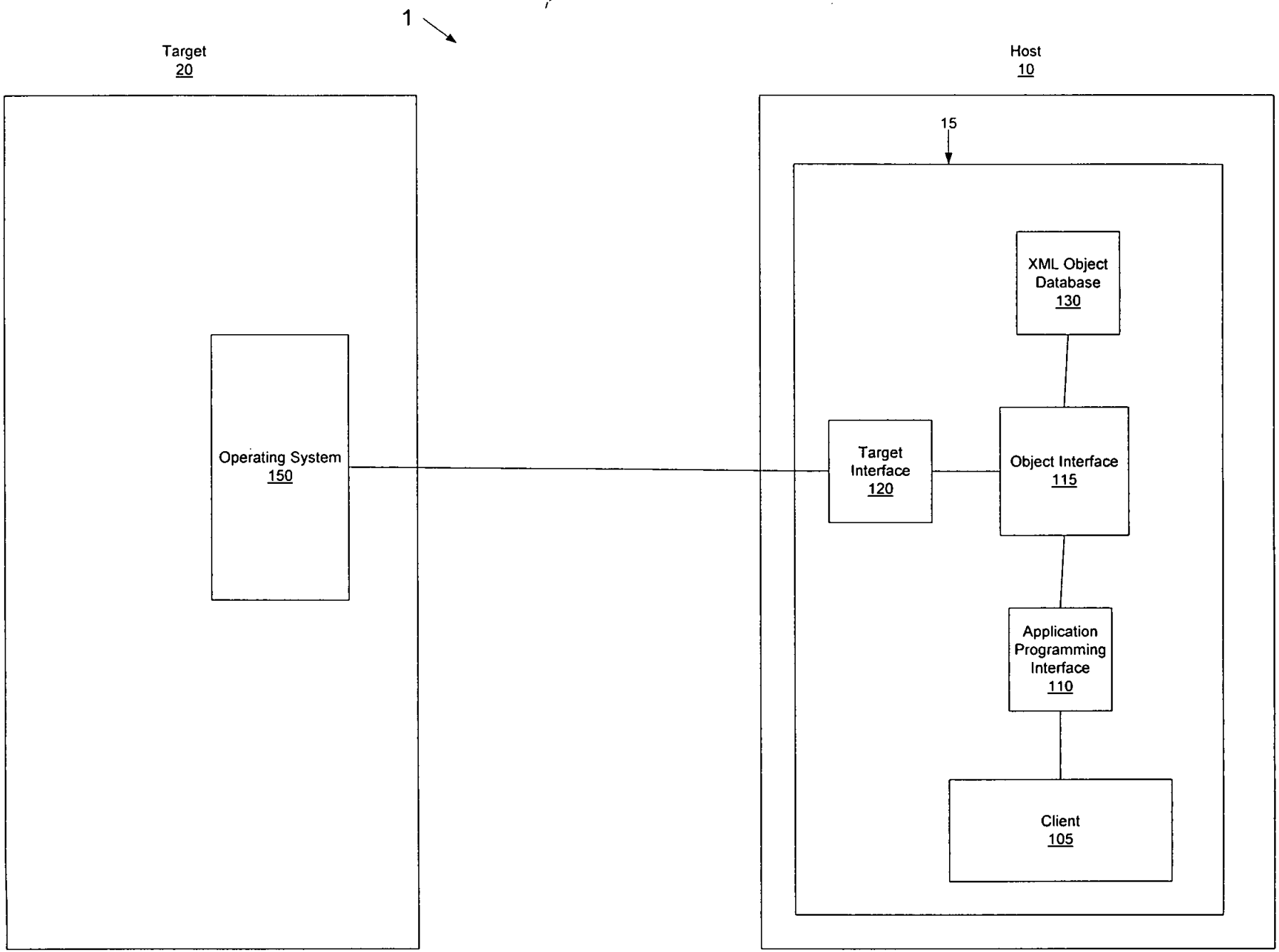
1
Target
20
Host
10
15
Operating System
150
XML Object
Database
130
Target
Interface
120
Object Interface
115
Application
Programming
Interface
110
Client
105

Fig. 2a

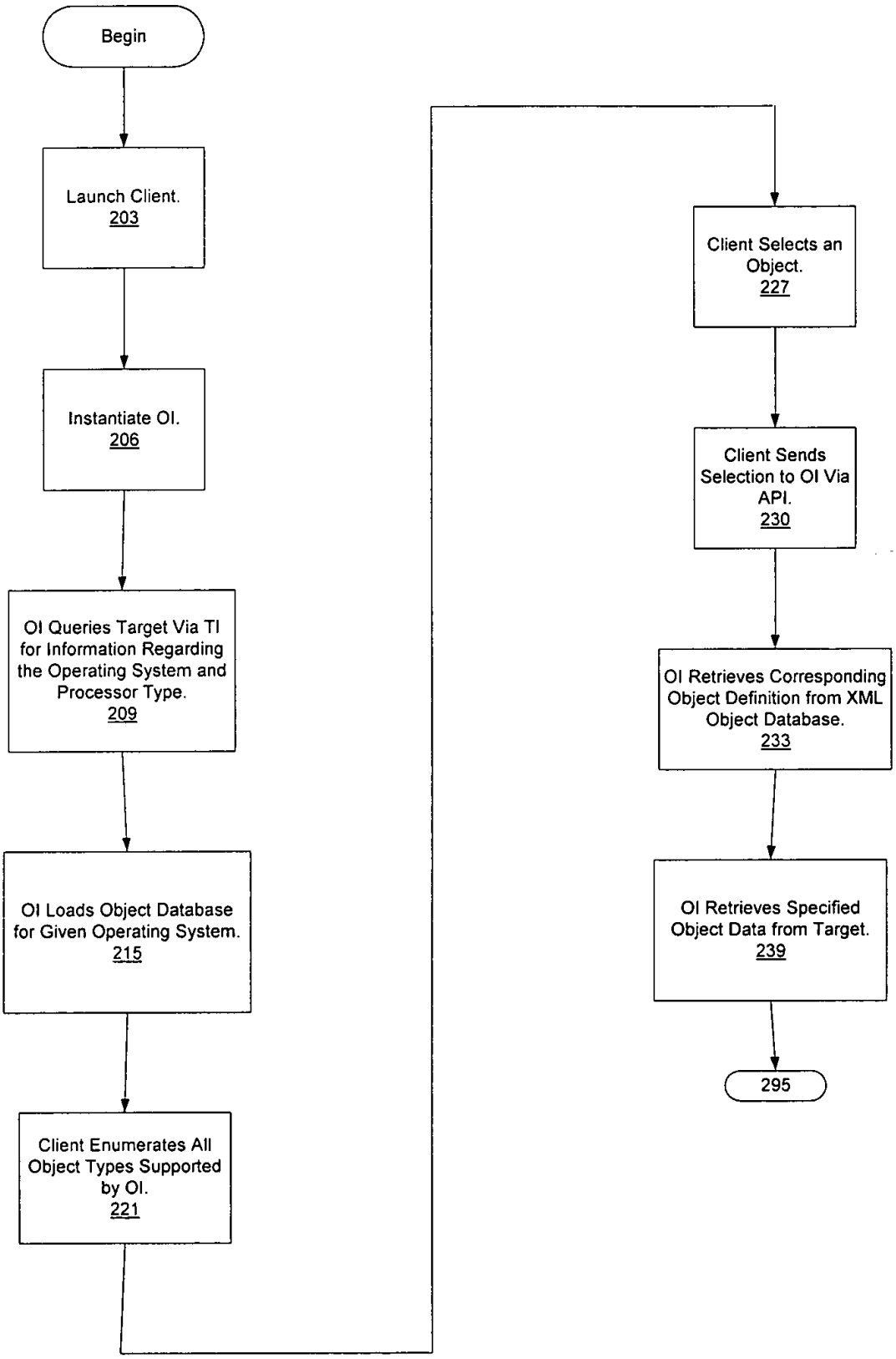
Begin
Launch Client.
203
Instantiate OI.
206
OI Queries Target Via TI for Information Regarding the Operating System and Processor Type.
209
OI Loads Object Database for Given Operating System.
215
Client Enumerates All Object Types Supported by OI.
221
Client Selects an Object.
227
Client Sends Selection to OI Via API.
230
OI Retrieves Corresponding Object Definition from XML Object Database.
233
OI Retrieves Specified Object Data from Target.
239
295

Fig. 2b

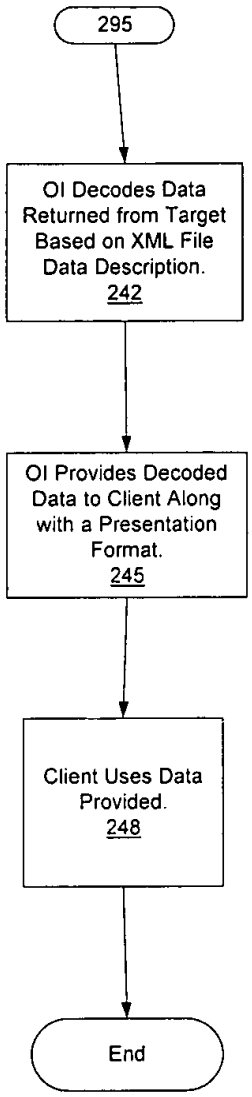
295
OI Decodes Data Returned from Target Based on XML File Data Description. 242
OI Provides Decoded Data to Client Along with a Presentation Format. 245
Client Uses Data Provided. 248
End

Fig. 3

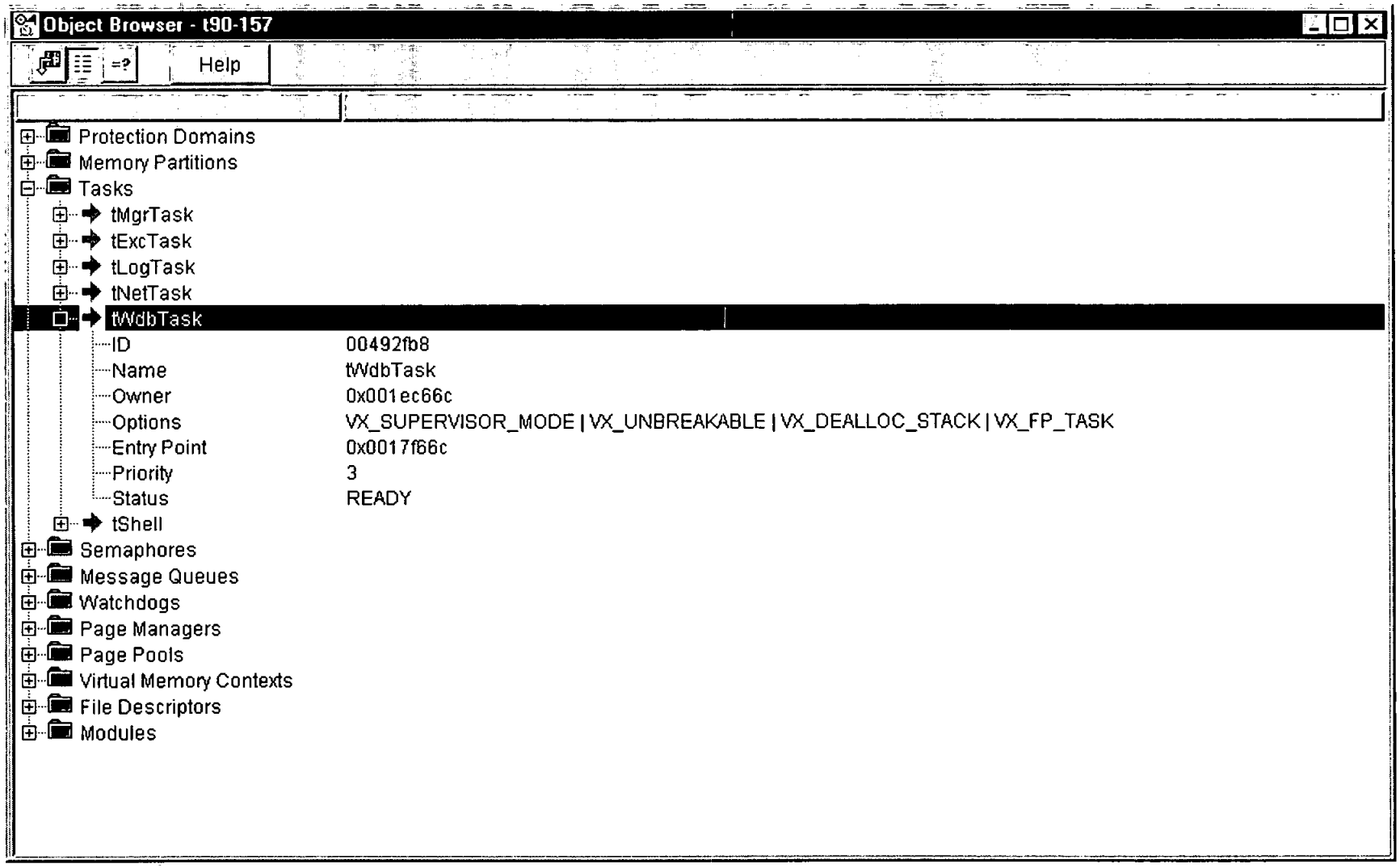
Object Browser - t90-157
Help
Protection Domains
Memory Partitions
Tasks
tMgrTask
tExcTask
tLogTask
tNetTask
tWdbTask
ID 00492fb8
Name tWdbTask
Owner 0x001ec66c
Options VX_SUPERVISOR_MODE | VX_UNBREAKABLE | VX_DEALLOC_STACK | VX_FP_TASK
Entry Point 0x0017f66c
Priority 3
Status READY
tShell
Semaphores
Message Queues
Watchdogs
Page Managers
Page Pools
Virtual Memory Contexts
File Descriptors
Modules

Fig. 4a

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE objType SYSTEM "objTypes.dtd">
<objType objTypeNumber="1" objTypeName="sem" objTypeHandler="tom"
symbolTableName="semClassId">
        <objTypeAttributes>
                <objTypeAttribute key="objTypeVisiblityLevel">
                        <value type="Integer">
                                <literal>1</literal>
                        </value>
                        <formatStr>
                                <literal>%d</literal>
                        </formatStr>
                </objTypeAttribute>
                <objTypeAttribute key="objTypeString1">
                        <valueStr>
                                <literal>Semaphore</literal>
                        </valueStr>
                </objTypeAttribute>
                <objTypeAttribute key="objTypeString2">
                        <valueStr>
                                <literal>Semaphores</literal>
                        </valueStr>
                </objTypeAttributc>
                <objTypeAttribute key="objTypeIconLarge">
                        <valueStr>
                                <literal>semL.ico</literal>
                        </valueStr>
                </objTypeAttribute>
                <objTypeAttribute key="objTypeIconSmall">
                        <valueStr>
                                <literal>sem.gif</literal>
                        </valueStr>
                </objTypeAttribute>
        </objTypeAttributes>
```

Fig. 4b

```
<objAttributes>
        <objAttribute key="objId">
                <label>
                        <literal>ID</literal>
                </label>
                <value>
                        <substitute fieldName="objId"/>
                </value>
                <formatStr>
                        <literal>%08x</literal>
                </formatStr>
                <detailLevel>
                        <literal>1</literal>
                </detailLevel>
        </objAttribute>
        <objAttribute key="objName">
                <label>
                        <literal>Name</literal>
                </label>
                <value>
                        <substitute fieldName="objName"/>
                </value>
                <formatStr>
                        <literal>%s</literal>
                </formatStr>
                <detailLevel>
                        <literal>1</literal>
                </detailLevel>
        </objAttribute>
        <objAttribute key="objOwner">
                <label>
                        <literal>Owner</literal>
                </label>
                <value>
                        <substitute fieldName="objOwner"/>
                </value>
                <formatStr>
                        <literal>0x%08x</literal>
                </formatStr>
                <detailLevel>
                        <literal>1</literal>
                </detailLevel>
        </objAttribute>
```

Fig. 4c

```
<objAttribute key="objHasChildren">
        <value>
                <switch fieldName="objChildListPtr">
                        <case caseValue="0">
                                <literal>false</literal>
                        </case>
                        <case caseValue="*">
                                <literal>true</literal>
                        </case>
                </switch>
        </value>
        <formatStr>
                <literal>%s</literal>
        </formatStr>
</objAttribute>
<objAttribute key="objType">
        <label>
                <literal>Type</literal>
        </label>
        <value>
                <substitute fieldName="semType"/>
        </value>
        <formatStr>
                <literal>0x%x</literal>
        </formatStr>
        <valueStr>
                <switch fieldName="semType">
                        <case caseValue="0">
                                <literal>Binary</literal>
                        </case>
                        <case caseValue="1">
                                <literal>Mutex</literal>
                        </case>
                        <case caseValue="2">
                                <literal>Counting</literal>
                        </case>
                        <case caseValue="*">
                                <literal>UNKNOWN</literal>
                        </case>
                </switch>
        </valueStr>
        <detailLevel>
                <literal>1</literal>
        </detailLevel>
</objAttribute>
```

Fig. 4d

```
<objAttribute key="objState">
    <!--There are two label entries here. The first will be overridden by the second,
if the second does not return a null value when processed. i.e. if the fieldMap contains an entry for the
"semType" fieldName, the second label will override the first when processed. In the case of a
getDatabaseAttributes() call, the second will return a null since "semType" will not be in the field map, and
the value contained in the first will persist.-->
    <label>
        <literal>State/Count/Owner</literal>
    </label>
    <label>
        <switch fieldName="semType">
            <case caseValue="0">
                <!--Binary Semaphore-->
                <literal>State</literal>
            </case>
            <case caseValue="1">
                <!--Mutex Semaphore-->
                <literal>Owner</literal>
            </case>
            <case caseValue="2">
                <!--Counting Semaphore-->
                <literal>Count</literal>
            </case>
            <case caseValue="*">
                <literal>Owner</literal>
            </case>
        </switch>
    </label>
    <value>
        <substitute fieldName="semState"/>
    </value>
    <formatStr>
        <switch fieldName="semType">
            <case caseValue="2">
                <!--Counting Semaphore - display as decimal-->
                <literal>%d</literal>
            </case>
            <case caseValue="*">
                <!--Any other sort of semaphore - display as hex-->
                <literal>0x%x</literal>
            </case>
        </switch>
    </formatStr>
```

Fig. 4e

```
<valueStr>
<!--Override the valueStr, even though the value and formatStr has
been specified. This allows decoding of the Semaphore statie into plain text. The only option is for a
Counting, or unknown Semaphore, where no CaseValue is specified. This allows the XML code to fall
through the Switch statement without finding a match. In this case, no value will be inserted into the
valueStr from here, leaving the XML code to compose one from the contents of the value combined with
the formatStr.-->
<switch fieldName="semType">
<case caseValue="0">
<!--Binary Semaphore-->
<switch fieldName="semState">
<case caseValue="0">
<literal>FULL</literal>
</case>
<case caseValue="*">
<literal>EMPTY</literal>
</case>
</switch>
</case>
<case caseValue="1">
<!--Mutex Semaphore-->
<switch fieldName="semState">
<case caseValue="0">
<literal>Unowned</literal>
</case>
</switch>
</case>
<!--For caseValue = 2 (Counting semaphore) or for any
other value, let this fall through without filling in a value. In this case, a valueStr will be composed
automatically from the value and formatStr entries-->
</switch>
</valueStr>
<detailLevel>
<literal>1</literal>
</detailLevel>
</objAttribute>
<objAttribute key="objOptions">
<label>
<literal>Options</literal>
</label>
<value>
<substitute fieldName="semOptions"/>
</value>
<formatStr>
<literal>0x%08x</literal>
</formatStr>
```

Fig. 4f

```
                <valueStr>
                        <typedef>
                                <switch fieldName="semOptions" fieldMask="0x0003">
                                        <case caseValue="0">
                                                <literal>SEM_Q_FIFO</literal>
                                        </case>
                                        <case caseValue="1">
                                                <literal>SEM_Q_PRIORITY</literal>
                                        </case>
                                        <case caseValue="*">
                                                <literal>SEM_Q_UNKNOWN</literal>
                                        </case>
                                </switch>
                                        <bitfield fieldName="semOptions"
                                                fieldMask="0xFFFFFFFC">
                                        <bit mask="0x0004">
                                                <literal>SEM_DELETE_SAFE</literal>
                                        </bit>
                                        <bit mask="0x0008">

<literal>SEM_INVERSION_SAFE</literal>
                                        </bit>
                                        <bit mask="*">
                                                <literal>SEM_UNKNOWN</literal>
                                        </bit>
                                </bitfield>
                        </typedef>
                </valueStr>
                <detailLevel>
                        <literal>1</literal>
                </detailLevel>
        </objAttribute>
        <objAttribute key="objIconSmall">
                <valueStr>
                        <switch fieldName="semState">
                                <case caseValue="0">
                                        <literal>sem.gif</literal>
                                </case>
                                <case caseValue="*">
                                        <literal>semempty.gif</literal>
                                </case>
                        </switch>
                </valueStr>
        </objAttribute>
</objAttributes>
```

Fig. 4g

```
<objGopher>
        <switch fieldName="cpu">
                <case caseValue="*">
                        <program>
                                <!--Default Get object info for sem - taken from PPC860-->
                                <![CDATA[! <+0x30*<+0x30@>>
<+0x28*$><+0x20*!><+0x8@> %nameLookup%<+0x34@b><+0x35@b><+0x48@>]]>
                        </program>
                        <tape>
                                <field fieldName="objId"/>
                                <field fieldName="objTypeNumber"/>
                                <field fieldName="objName"/>
                                <field fieldName="objOwner"/>
                                <field fieldName="objChildListPtr"/>
                                <field fieldName="semType"/>
                                <field fieldName="semOptions"/>
                                <field fieldName="semState"/>
                        </tape>
                </case>
        </switch>
</objGopher>
<enumByType>
        <switch fieldName="cpu">
                <case caseValue="*">
                        <program>
                                <![CDATA[*+0x38** -0x18! <+0x30*<+0x30@>>
<+0x28*$><+0x20*!><+0x8@> <+0x34@b><+0x35@b><+0x48@>+0x18@]]>
                        </program>
                        <tape>
                                <field fieldName="objId"/>
                                <field fieldName="objTypeNumber"/>
                                <field fieldName="objName"/>
                                <field fieldName="objOwner"/>
                                <field fieldName="objChildListPtr"/>
                                <field fieldName="semType"/>
                                <field fieldName="semOptions"/>
                                <field fieldName="semState"/>
                                <field fieldName="objSiblingPtr"/>
                        </tape>
```

Fig. 4h

```
                                        <program>
                                                <![CDATA[%objSiblingPtr% -0x18! <+0x30*<+0x30@>>
<+0x28*$><+0x20*!><+0x8@><+0x34@b><+0x35@b><+0x48@>+0x18@]]>
                                        </program>
                                        <tape>
                                                <field fieldName="objId"/>
                                                <field fieldName="objTypeNumber"/>
                                                <field fieldName="objName"/>
                                                <field fieldName="objOwner"/>
                                                <field fieldName="objChildListPtr"/>
                                                <field fieldName="semType"/>
                                                <field fieldName="semOptions"/>
                                                <field fieldName="semState"/>
                                                <field fieldName="objSiblingPtr"/>
                                        </tape>
                                </case>
                        </switch>
                </enumByType>
</objType>
```

Fig. 5

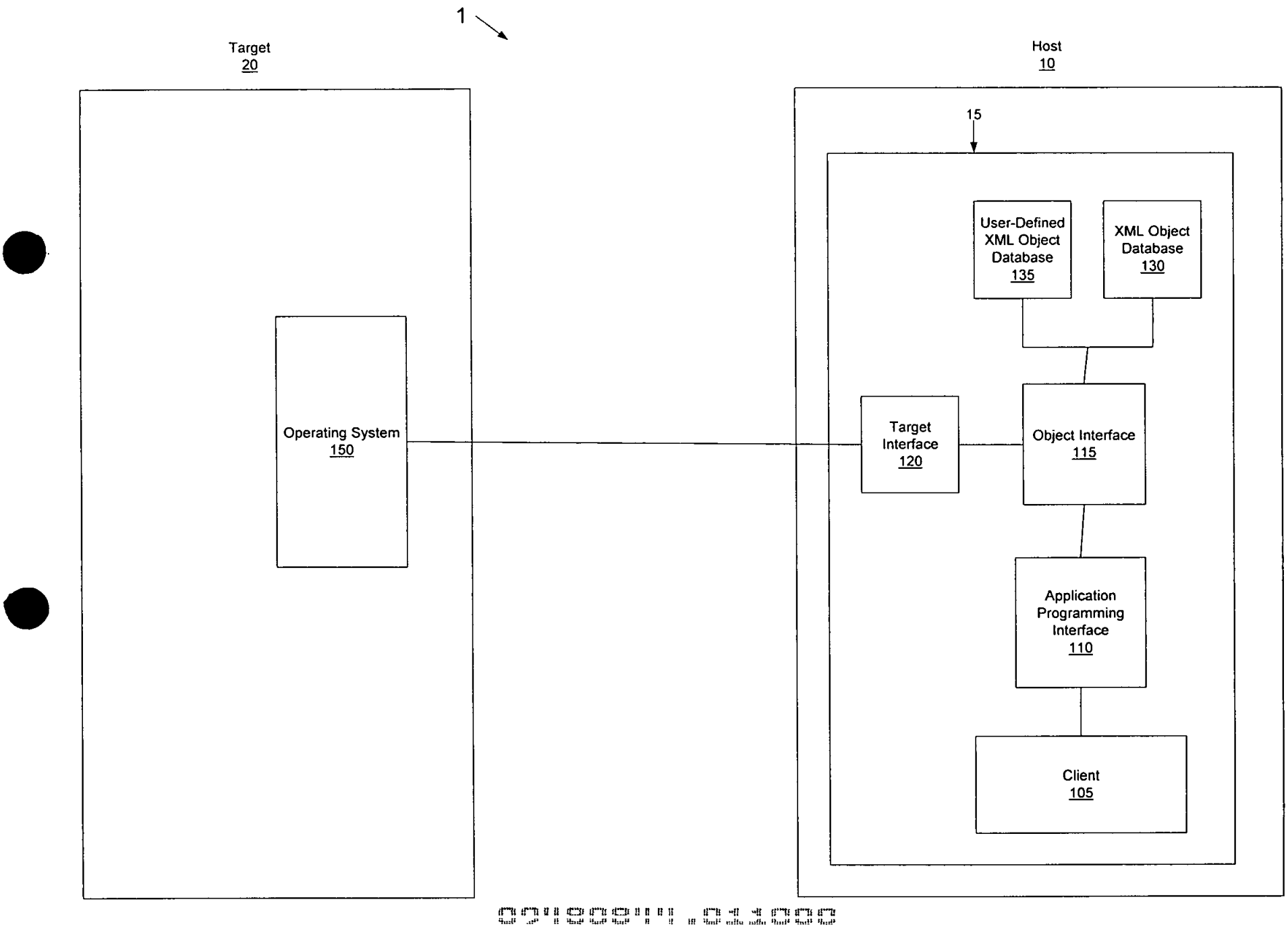
1
Target
20
Host
10
15
User-Defined XML Object Database
135
XML Object Database
130
Operating System
150
Target Interface
120
Object Interface
115
Application Programming Interface
110
Client
105

Fig. 6a

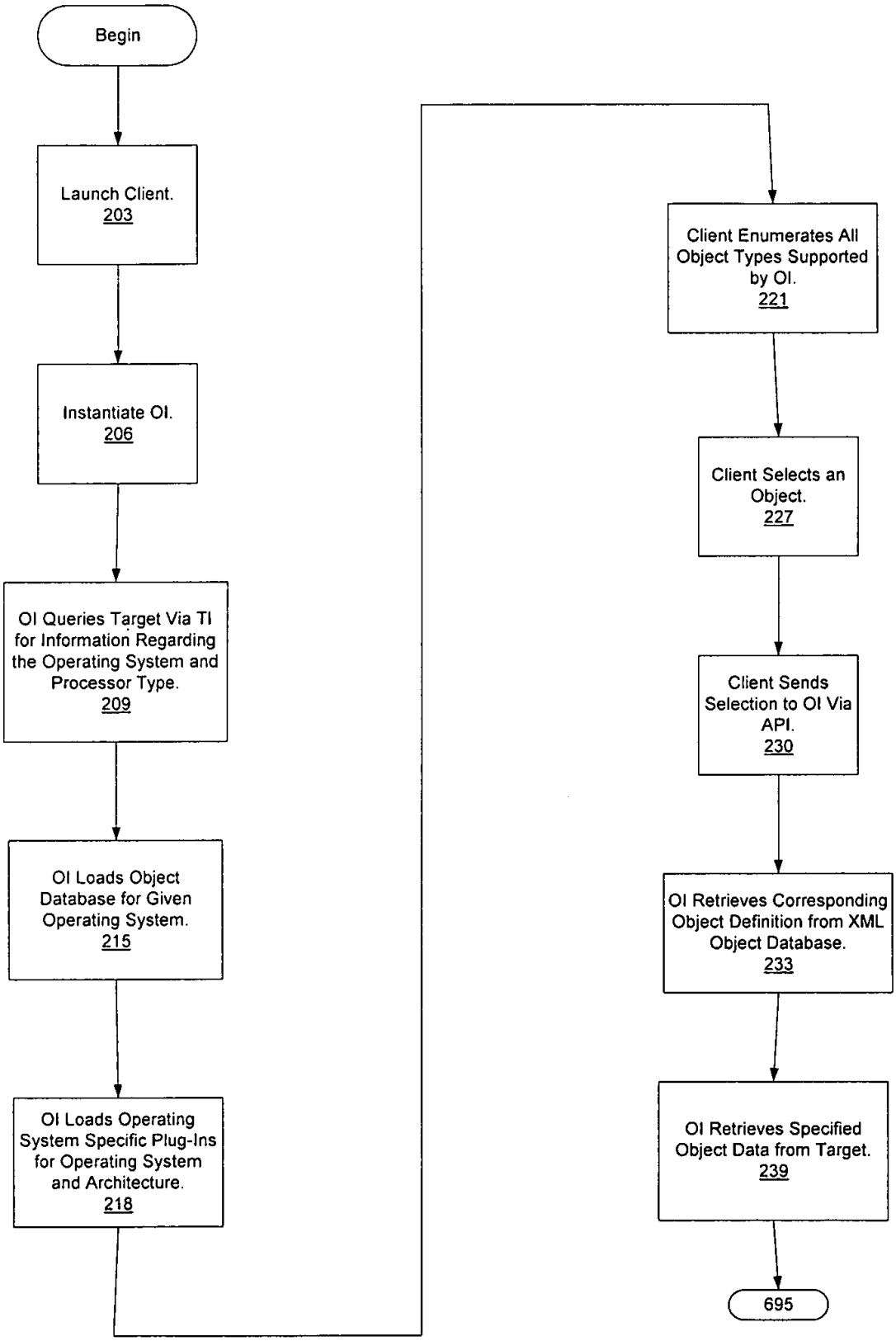
Begin
Launch Client.
203
Instantiate OI.
206
OI Queries Target Via TI for Information Regarding the Operating System and Processor Type.
209
OI Loads Object Database for Given Operating System.
215
OI Loads Operating System Specific Plug-Ins for Operating System and Architecture.
218
Client Enumerates All Object Types Supported by OI.
221
Client Selects an Object.
227
Client Sends Selection to OI Via API.
230
OI Retrieves Corresponding Object Definition from XML Object Database.
233
OI Retrieves Specified Object Data from Target.
239
695

Fig. 6b

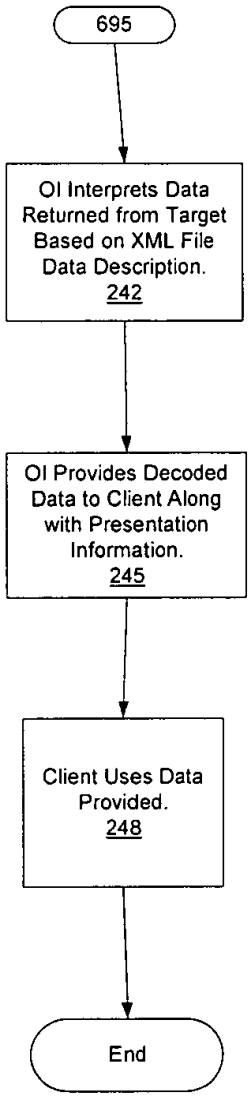
695
OI Interprets Data Returned from Target Based on XML File Data Description.
242
OI Provides Decoded Data to Client Along with Presentation Information.
245
Client Uses Data Provided.
248
End

Fig. 7

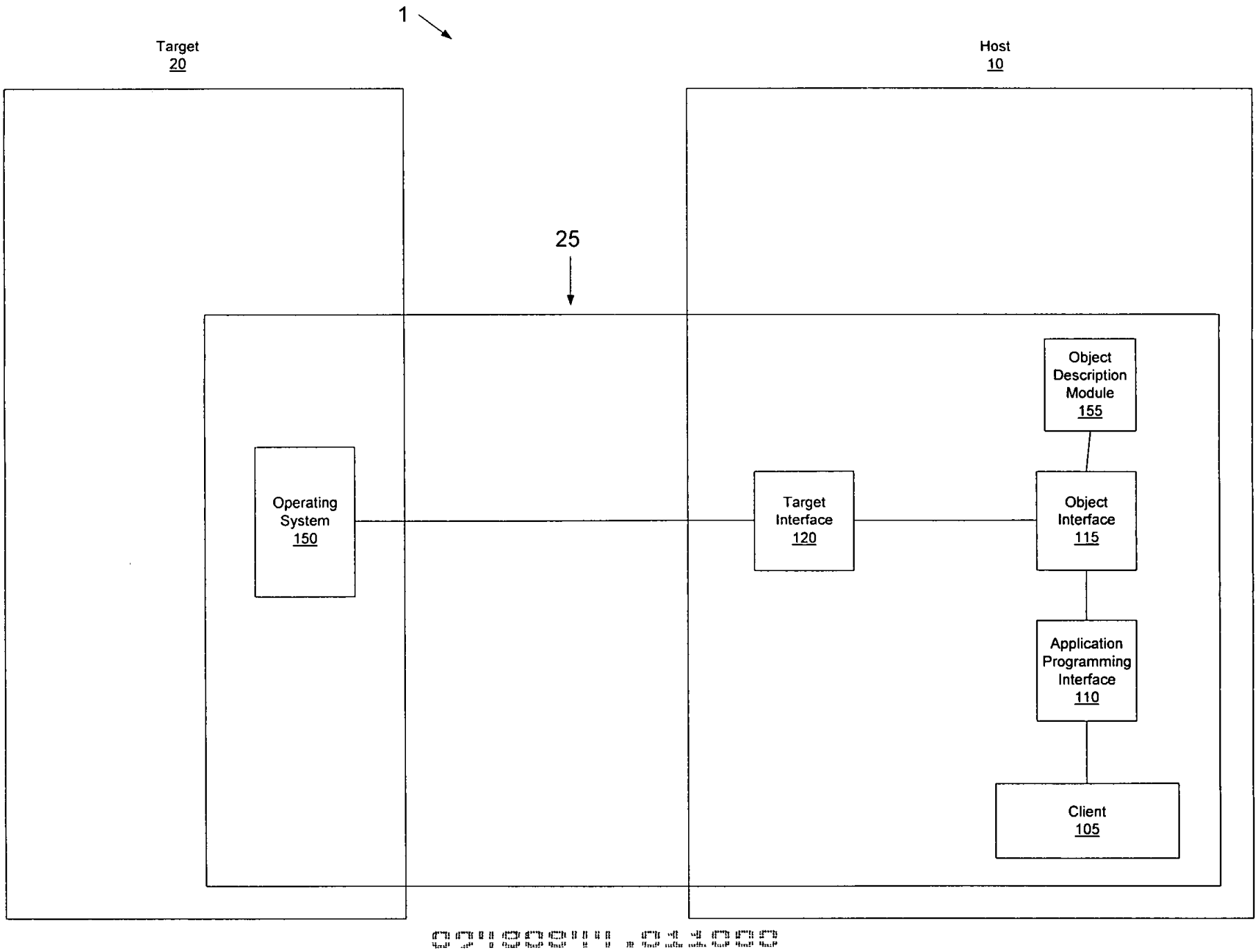
1
Target
20
Host
10
25
Object Description Module
155
Operating System
150
Target Interface
120
Object Interface
115
Application Programming Interface
110
Client
105

Fig. 8

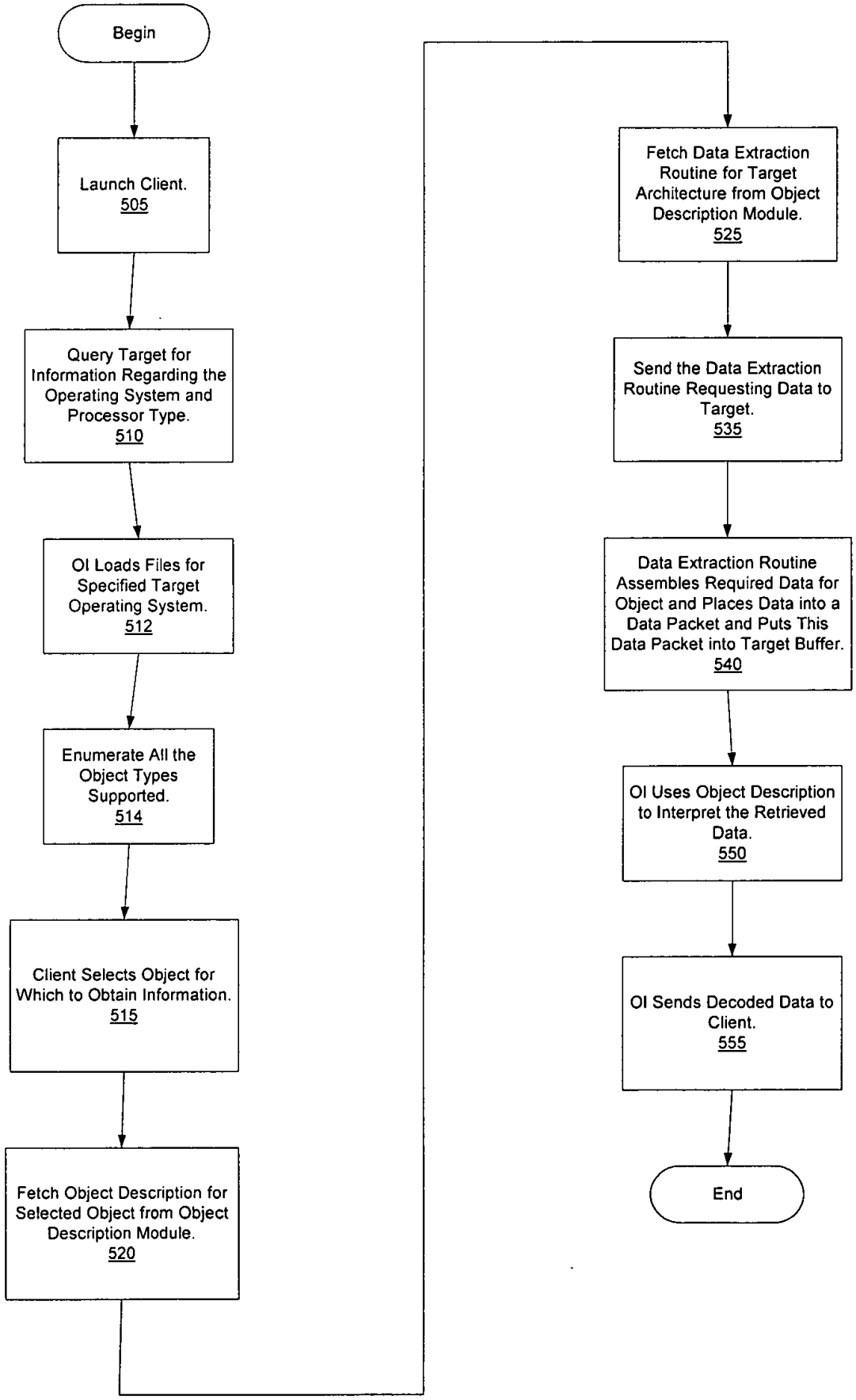
Begin
Launch Client. 505
Query Target for Information Regarding the Operating System and Processor Type. 510
OI Loads Files for Specified Target Operating System. 512
Enumerate All the Object Types Supported. 514
Client Selects Object for Which to Obtain Information. 515
Fetch Object Description for Selected Object from Object Description Module. 520
Fetch Data Extraction Routine for Target Architecture from Object Description Module. 525
Send the Data Extraction Routine Requesting Data to Target. 535
Data Extraction Routine Assembles Required Data for Object and Places Data into a Data Packet and Puts This Data Packet into Target Buffer. 540
OI Uses Object Description to Interpret the Retrieved Data. 550
OI Sends Decoded Data to Client. 555
End

Fig. 9a

```
<?xml version="1.0" encoding"utf-8"?>

<object name="binarySemaphore"
icon="k:\wpwr\host\resource\bitmaps\WindView\events\semBCreate.bmp">

        <helpText>

                <synopsis>
                        This is a short description of the object.
                </synopsis>

                <description>
                        This is a much longer description of the object.
                        This description can contain anything the user desires.
                <description>

        </helpText>

        <publicDataGet>

                <helpText>
                        <synopsis>Get all public data members</ synopsis>
                        <description> A full description of the routine</description>
                </help Text>

                <requestBegin>
                        <call name="semRequestBegin"/>
                        <return type="UINT8*"/>
                        <parm type="SEM_ID"name="semId"/>
                </requestBegin>

                <requestEnd>

                        <callname="semRequestEnd"/>
                        <return type= "STATUS"/>
                        <parm type="UINT8 *" name="pBuff"/>
                </requestEnd>
```

Fig. 9b

```
<data>
    <dataItem type="UINT"idref="pkLength"display="never"/>
    <dataItemtype="UINT"idref="semId"
        text="Handle"format="0x08x"display="always"
        position="1"/>
    <dataItem type="UINT"idref="semClassId"display="never"/>
    <dataItem type="string" idref="name"
        text="Name" format="%s" display="always"
        position="0"/>
    <dataItem type="string" idref="owner"
        text="Owner" format="%s" display="always"
        position="2"/>
    <dataItem type="int"diref="value"
        text="State"format="translate"display="always"
        position="3"
        <translate key ="0" value="Empty" format="%s"/>
        <translate key="1" value="Taken" format="%s"/>
        <translate key="*"value="Unknown" format="%s"/>
    </dataItem>

</data>

</pubicDataGet>

</object>
```